# UNITED STATES PATENT AND TRADEMARK OFFICE

**UNITED STATES DEPARTMENT OF COMMERCE**
**United States Patent and Trademark Office**
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

| APPLICATION NO. | FILING DATE | FIRST NAMED INVENTOR | ATTORNEY DOCKET NO. | CONFIRMATION NO. |
|---|---|---|---|---|
| 09/930,541 | 08/14/2001 | Leonid M. Arbouzov | SUNMP014 | 1706 |

25920 7590 08/18/2004

MARTINE & PENILLA, LLP
710 LAKEWAY DRIVE
SUITE 170
SUNNYVALE, CA 94085

| EXAMINER | |
|---|---|
| INGBERG, TODD D | |

| ART UNIT | PAPER NUMBER |
|---|---|
| 2124 | |

DATE MAILED: 08/18/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

| Office Action Summary | Application No. | Applicant(s) | |
|---|---|---|---|
| | 09/930,541 | ARBOUZOV ET AL. | |
| | Examiner | Art Unit | |
| | Todd Ingberg | 2124 | |

*-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --*

**Period for Reply**

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

**Status**

1)☒ Responsive to communication(s) filed on *09 January 2002*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

**Disposition of Claims**

4)☒ Claim(s) *1-20* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-20* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

**Application Papers**

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *09 January 2002* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

**Priority under 35 U.S.C. § 119**

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

**Attachment(s)**

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

## DETAILED ACTION

Claims 1 – 20 have been examined.

### *Drawings*

1. The drawings were received on January 9, 2002. These drawings have been entered.

2. Figure 4 is objected to because #406 has the word "Program" written "PRogram".

### *Priority*

3. Applicant's claim for domestic priority ( 60/291,670) under 35 U.S.C. 119(e) is acknowledged.

### *Information Disclosure Statement*

4. The Information Disclosure Statement filed on August 8, 2001 has been considered.

### *Claim Rejections - 35 USC § 101*

5. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

6. Claims 1 – 20 are rejected on the grounds of being for non statutory. A simple amendment will overcome this rejection.

Claim 1

A method **on a computer readable medium** for preprocessing Java application code, comprising the operations of:

receiving a Java template file, the Java template file including Java programming language code and meta code;

processing the Java template to create an intermediate program using the meta code, wherein the intermediate program is a Java program;

compiling the intermediate program to create an intermediate, class, wherein the intermediate class is a Java based class; and

generating an object text file using the intermediate class.

Claim 8

A method **on a computer readable medium** for transforming lines of a Java template for preprocessing, comprising the operations of:

obtaining a line of text from a Java template;

determining if the line of text begins with a meta symbol, wherein the meta symbol is a predefined symbol indicating the line of text is a preprocessor directive;

removing the meta symbol from the line of text when the line of text begins with the meta symbol; and

transforming the line of text into a function argument when the line of text does not begin with the meta symbol.

Claim 16

A Java preprocessor **on a computer readable medium** , comprising:

a meta code converter capable of processing a Java template to create an intermediate program using meta code included in the Java template; and

an object text generator that compiles the intermediate program to create an intermediate class, wherein the intermediate class is a .lava based class, the object text generator also capable of generating an object text filed using the intermediate class.

***Knowledge of the Ordinary Artisan***

7. The following information should be known to one of ordinary skill in the art at the time of invention.

The basic workings of the C++ preprocessor - The C++ preprocessor reads the file and locates the directives in the C++ file. Directives begin the line with a "#". The following are C++ directives and are located on pages 424 to 425 of the Design and Evolution of C++ book written

by the inventor of C++, Bjarne Stroustrup. This book is a historical perspective of design decisions he made in the development of the programming language.

# include

- Make interface definition available.
- Compose source text

#define

- Define symbolic constants
- Define open subroutines
- Define generic subroutines
- Define generic "types"
- Renaming
- String concatenation
- Define special purpose syntax
- General macro processing

#ifdef

- version control
- Commenting out code

#pragma

- Control of layout
- Informing the compiler about unusual control flow

The result of the preprocessor is a file ready for compilation by the C++ compiler. This does not guarantee the code included or generates will pass the syntax of the C++ compiler. The syntax is dependent on the correctness of the code written to include/expand. The directive "#" is removed because it's purpose is for the preprocessor to identify the present of a directive, the compiler does not recognize the directives and will provide a syntax error.

***Term Interpretations***

8. The following terms from the Specification are given the following interpretations

A. ***meta language*** – This refers to the JAVA programming language.

B. ***header file*** - This term in C++ means a file typically containing source code. The interpretation in this application means a file containing a PROLOG (page 10 of Specification). A PROLOG is a documentation area where programmers document the programs authors and history. In short it is a file containing comments. The content of the files #included are not given patentable weight.

C. ***footer files*** – This term is being used by Applicant for what is usually referred to as the header file in C++. A file containing source code such as class files (page 10 of Specification). The content of the files #included are not given patentable weight.

D. ***intermediate program*** – the output from the preprocessor ready to be used as input to the JAVA compiler.

E. ***appropriate instructions*** – depends on the operation defined in the directive. If it is a macro appropriate instructions are the results of the macro expansion.

F. ***meta code*** – preprocessor file – input to the preprocessor. Containing directives with or without arguments and JAVA code.

G. ***meta symbol*** – The actual indicator that this line is meant to be processed by the preprocessor. Also, formally called a directive. In C and C++ a directive is indicated by a "#".

***Claim Rejections - 35 USC § 103***

9. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

Claims 1 – 20 are rejected under 35 U.S.C. 103(a) as being unpatentable over **JAVA** the programming language as taught by the text book JAVA Primer Plus" by Paul M. Tyma et al published March 6, 1996. Makes specific mention that JAVA initially did not have a preprocessor (pages 256 – 257) but the programming languages C and C++ do have a preprocessor in view the ANSI standard programming language **C++** as taught by the text book "The Annotated C++ Reference Manual" (ANSI Base Document) written by Margaret Ellis and the inventor of C++ Bjarne Stroustrup and published June 7, 1990.

**Motivation to Combine References**

Programming languages such as ANSI standard C and ANSI standard C++ have preprocessors. Preprocessors enable a programming language to have such features as file inclusion (C++, page 375), conditional compilation, (C++, page 376), Error Directives (C++, page 378) and macro substitution (C++ page 369). Therefore, it would have been for one of ordinary skill in the art at the time of invention to add a preprocessor like the one in C++ for well over a decade prior to invention to the JAVA programming language because a preprocessor adds the capability for a programming language to perform " … macro substitution, conditional compilation, and inclusion of named files." (C++, page 369).

**Claim 1**

A method for preprocessing Java application code, comprising the operations of:

receiving a Java template file, the Java template file including Java programming language code and meta code;

processing the Java template to create an intermediate program using the meta code, wherein the intermediate program is a Java program;

compiling the intermediate program to create an intermediate, class, wherein the intermediate class is a Java based class; and

generating an object text file using the intermediate class.

**Examiner's Response**

The claim is for the adding a preprocessor to the programming language that support the #include function found in C++. The basic workings of a preprocessor are described with examples in C++ on pages 369 – 370, 372 – 373 and 375 – 376. The ability to include a class file in C++ to become an object is part of the C++ programming language ( see C++ [age 374 ## operator). It is deemed obvious to one of ordinary skill in the art at the time of invention to add a preprocessor to the JAVA programming language.

**Claim 2**

A method as recited in claim 1, wherein the meta code can include Java programming language statements.

**Examiner's Response**

The examples in C++ contain examples where directive contain C++ code ( C++, page 372) and do not contain C++ code (page 376 – 377 conditional compilation directives).

**Claim 3**

A method as recited in claim 2, wherein the meta code is equivalent to the Java programming language.

**Examiner's Response**

In the examples such as on page 372 where C++ code is the output of the preprocessor this is inherent need because the next phase is to compile the C++ code with the compiler. The code must be the target language of the compiler. For the JAVA preprocessor to generate JAVA code for the JAVA compiler is equivalent.

**Claim 4**

A method as recited in claim 1, further comprising the operation of transforming lines in the Java template.

**Examiner's Response**

In the examples such as on page 375 ( the #include) where C++ code is the output of the preprocessor this is inherent need because the next phase is to compile the C++ code with the compiler. The code must be the target language of the compiler. For the JAVA preprocessor to generate JAVA code for the JAVA compiler is equivalent.

**Claim 5**
A method as recited in claim 4, wherein the transformed lines are entered into the intermediate program.
**Examiner's Response**
Both claim 3 and 4 result in C++ code ready for compilation. This is the intermediate program.

**Claim 6**
A method as recited in claim 5, further comprising the operation of entering header data into the intermediate program.
**Examiner's Response**
The #include directive as per above is a file copy into a program. The header file is a PROLOG. No patentable weight is given to the content of a file to be included.

**Claim 7**
A method as recited in claim 5, further comprising the operation of entering footer data into the intermediate program.
**Examiner's Response**
The #include directive as per above is a file copy into a program. The footer file is program code. No patentable weight is given to the content of a file to be included.

**Claim 8**
A method for transforming lines of a Java template for preprocessing, comprising the operations of:
obtaining a line of text from a Java template;
determining if the line of text begins with a meta symbol, wherein the meta symbol is a predefined symbol indicating the line of text is a preprocessor directive;
removing the meta symbol from the line of text when the line of text begins with the meta symbol; and
transforming the line of text into a function argument when the line of text does not begin with the meta symbol.
**Examiner's Response**
The claim is for the adding a preprocessor to the programming language that support the #define function found in C++. (Note: the # is the meta symbol). The claim is for the ability to define a macro with a directive and the ability to expand the macro. The macro is identified by the # at the beginning of the line of code and expanded by the preprocessor. The preprocessor produces an output file which does not contain the directives. Directives are not recognized in the next step of compiling the source code by a compiler. Pages 372 to 373of the C++ reference teaches the definition and expansion of macros. The #define directive is used to perform macro definition. The example shows the before and after of a macro expansion. Note the # directive symbol is not present in the expanded macro. The modification to JAVA to add a preprocessor to provide the ability of defining and expanding macros is deemed obvious to one of ordinary skill in the art at the time of invention.

**Claim 9**
A method as recited in claim 8, wherein transforming the line of text into a function argument comprises the operation of enclosing the line of text in quotes.
**Examiner's Response**
In the example of claim 8 note the symbol SIDE was used as an argument in defining the size of a multi dimension array (chessboard).

**Claim 10**
A method as recited in claim 9, wherein transforming the line of text into a function argument further comprises the operation of prepending the line of text with an internal preprocessor method call and an open bracket.
**Examiner's Response**
Page 373 shows an example of a macro expanding into a function. Note the function extract and the parameter list.

**Claim 11**
A method as recited in claim 10, wherein transforming the line of text into a function argument further comprises the operation of appending a closing bracket and a semicolon to the end of the line of text.
**Examiner's Response**
The example of claim 10 shows the argument list has both brackets and a semicolon terminating the line as needed to be syntactically correct for the next phase of compiling the code

**Claim 12**
A method as recited in claim 11, wherein the quotes are double quotes.
**Examiner's Response**
The use of double quotes to identify a macro is not given patentable weight. The need to distinguish a macro is inherent in writing a preprocessor that supports macros.

**Claim 13**
A method as recited in claim 8, further comprising the operation of determining whether macros are present in the line of text.
**Examiner's Response**
The C++ preprocessor uses the "#define identifier …" as depicted on page 372.

**Claim 14**
A method as recited in claim 13, further comprising the operation of transforming macros present in the line of text into appropriate instructions.
**Examiner's Response**
Both the examples of macro symbol expansion and macro function expansion are examples of transforming into appropriate code.

**Claim 15**

A method as recited in claim 8, further comprising the operation of writing header data, footer data, and the transformed lines of text to an intermediate program.

**Examiner's Response**

Page 375 of C++ teaches the including of files with the #include filename. The content of the files weather it is comments (PROLOG) of code the affect of the include operation is the same in the preprocessor.

**Claim 16**

A Java preprocessor, comprising:

a meta code converter capable of processing a Java template to create an intermediate program using meta code included in the Java template; and

an object text generator that compiles the intermediate program to create an intermediate class, wherein the intermediate class is a java based class, the object text generator also capable of generating an object text filed using the intermediate class.

**Examiner's Response**

The claim is for the adding a preprocessor to the programming language that support the #include function found in C++ that includes files containing a class(es). Classes are the definitions for objects. C++ is an object oriented programming language. The ability to have a directive to support defining and calling classes is taught in C++ on page 374 with the ## Operator.

**Claim 17**

A Java preprocessor as recited in claim 16, wherein the meta code can include Java programming language statements.

**Examiner's Response**

The examples in C++ contain examples where directive contain C++ code ( C++, page 372) and do not contain C++ code (page 376 – 377 conditional compilation directives).

**Claim 18**

A Java preprocessor as recited in claim 17, wherein the meta code is equivalent to the Java programming language.

**Examiner's Response**

In the examples such as on page 372 where C++ code is the output of the preprocessor this is inherent need because the next phase is to compile the C++ code with the compiler. The code must be the target language of the compiler. For the JAVA preprocessor to generate JAVA code for the JAVA compiler is equivalent.

**Claim 19**

A Java preprocess :as recited in claim 18, further comprising a preprocessor library interface module that specifies a contract that can be extended by any Java preprocessor library that is to be supported by the Java preprocessor.

**Examiner's Response**

The limitation "any Java preprocessor library" is given the interpretation of any library able to be accessed by the preprocessor. Include directives in C++ as on pages 375 to 376 of C++

shows different ways to point to files to be included. The #include "filename" searches directories such as libraries for files to be included.

**Claim 20**

A Java preprocessor as recited in claim 19, further comprising an executor module that calculates the name of the object text generator class and invokes main method of the object text generator class.

**Examiner's Response**

C++ is an object oriented programming language where the #include allows the including of class files. Classes are well known in object technology to define objects. The claim is interpreted to be the processing of the class files in C++ to be an equivalent function for the JAVA programming language.

## *Conclusion*

10. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

A. Design and Evolution of C++ book written by the inventor of C++, Bjarne Stroustrup. This book is a historical perspective of design decisions he made in the development of the C++ programming language.

B. "JAVA! " by Tim Richey published September 22, 1995 . Teaches JAVA language constructs and environment.

## *Correspondence Information*

11. Any inquiry concerning this communication or earlier communications from the examiner should be directed to **Todd Ingberg** whose telephone number is (703) 305-9775. The examiner can normally be reached during the following hours:

| Monday | Tuesday | Wednesday | Thursday | Friday |
|---|---|---|---|---|
| 6:15 – 1:30 | 6:15- 3:45 | 6:15 – 4:45 | 6:15-3:45 | 6:15-130 |

This schedule began December 1, 2003 and is subject to change.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, **Kakali Chaki** can be reached on (703) 305-9662. Please, note that as of August 4, 2003 the **FAX number** changed for the organization where this application or proceeding is assigned is **(703) 872-9306**.

Also, be advised the United States Patent Office **new address** is

Post Office Box 1450
Alexandria, Virginia 22313-1450

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the receptionist whose telephone number is (703) 305-9700.

**Todd Ingberg**
Primary Examiner
Art Unit 2124
July 25, 2004